ARCHITECTURE OF METHOD FOR FETCHING MICROPROCESSOR'S INSTRUCTIONS

FIELD OF THE INVENTION

[0001] This invention relates to a kind of architecture of method for fetching microprocessor's instructions, in which the method is supposed to read two instructions in program memory in the event of a conditional branch, otherwise read an instruction only for reducing power consumption.

BACKGROUND OF THE INVENTION

[0002] As a computer's effectiveness might be commented on the basis of processing time of an instruction, the way leading to cut down the processing time of instruction is a great issue pending further improvements.

[0003] A single-cycle instruction is an instruction that can be executed and completed within a cycle and allow a microprocessor to pre-read the next instruction, however, as a matter of fact, it isn't the case that all the instructions in a program are single-cycle instructions.

[0004] When executing the general logic instructions shown in Fig. 1, a microprocessor is supposed to have run an instruction and consecutively pre-read the next one totally in an instruction cycle by adding value 1 to a program counter (PC). If the next instruction is a "CALL" instruction, the PC will be added with a discrete variable "M" instead of the usual 1 to make the PC value discontinuous. Before so doing, the program would need a no operation (NOP) instruction for loading the correct address of the variable "M" to the PC for fetching and executing the instruction called. Such a makeshift requires at least one more instruction cycle that usually deteriorates the microprocessor's effectiveness.

[0005] As to improve abovesaid defect, the procedure of an existing method for fetching instructions shown in Fig. 2 is to pre-read instructions at address N+1 and N+2 in the meanwhile the instruction at address N is executed, and at this moment, the method also decodes the N+1 instruction. In case the N+1 instruction is found not a general logic instruction, such as a "CALL" instruction for example, the next instruction to be executed will be replaced by a "NOP" instruction for loading the correct address of the variable "M" and pre-reading the instructions at address "M" and "M"+1, so that the called instruction at address "M" will be executed in the next instruction cycle for waiving of the abovesaid extra cycle to thereby heighten the processing effectiveness.

[0006] Nevertheless, more power is consumed during the process of fetching and storing those two instructions previously, and it is considered still rooms available for improvement of power consumption.

SUMMARY OF THE INVENTION

[0007] The primary object of this invention is to provide a kind of architecture of method for fetching microprocessor's instructions. The method which normally pre-reads a next instruction would pre-read and pre-decode two next instructions in case it encounters a "CALL" instruction so as to waive unnecessary reading of program memory and reduce power consumption accordingly.

[0008] Another object of this invention is to provide a kind of architecture of method for fetching microprocessor's instructions. In the process of executing instructions, a processing unit is employed to decode an instruction next to the current one for setting the state of an instruction reading-amount register. In the case the next instruction is found a conditional branch instruction, both an odd and an even address buffer register are enabled simultaneously for fetching two next

instructions, wherein the choice of an immediate one is determined by the processing unit. Then only one of the address buffer register is enabled for fetching an instruction in order to waive any unnecessary reading of program memory for reducing power consumption.

[0009] For more detailed information regarding advantages or features of this invention, at least an example of preferred embodiment will be fully described below with reference to the annexed drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] The related drawings in connection with the detailed description of this invention to be made later are described briefly as follows, in which:

Fig. 1 is a schematic view of timing chart of a conventional method for fetching microprocessor's instructions;

Fig. 2 is another schematic view of timing chart of a conventional method for fetching microprocessor's instructions;

Fig. 3 is a flowchart of a method of this invention for reading microprocessor's instructions;

Fig. 4 is an embodiment of the method of this invention for fetching microprocessor's instructions; and

Figs. 5 and 6 are timing charts of the method of this invention for fetching microprocessor's instructions.

DETAILED DESCRIPTION OF THE INVENTION

[0011] In general, a program's instructions of computer might be divided into four categories: the general instructions as a first category for execution of general logic instructions; the unconditional branch instructions as a second category; the "CALL" and the "RETURN" instructions as a third category; and the conditional

branch instructions as a fourth category.

[0012] A next instruction succeeding to the current one might have several alternatives, including: an only address made by adding 1 to the present PC (program counter) value of the first category; or, a new and only address contained in the current instruction of the second category; or, a new and only address contained in the current instruction of the third category, in which a return address can be found in the stack; or, an address at PC+1 or PC+2 of the fourth category, which is to be determined by a processing unit. Therefore, when a succeeding instruction is decoded as a conditional branch instruction, a method of this invention for fetching microprocessor's instructions is supposed to pre-read and pre-decode two sequential instructions and choose to execute one of those alternatives.

[0013] As shown in Fig. 3, in running a program, the method of this invention shall enter to choose one of four options after execution of a buffer step **301** and a pre-reading step **302**. If the next instruction is decoded and found a general logic instruction **303** for example, the procedure of this method is to add 1 to a PC (namely, PC+1) **307** and set an instruction reading-amount register in a state for fetching a next instruction only **311**; or, if it is found an unconditional branch instruction **304**, the PC will point to a new address **308** and set the instruction reading-amount register in a state for fetching an instruction **311**; or, if it is found a "CALL" or a "RETURN" instruction **305**, the PC will point to a new address **309** and set the instruction reading-amount register in a state for fetching a specified instruction only **311**, or if it is found a conditional branch instruction **306**, the PC will point to a next then a further next address **310** (PC+2) and set the instruction reading-amount register in a state for fetching two instructions **312** for the

processing unit to choose and execute one of the alternatives **313**.

[0014] Fig. 4 is a kind of architecture embodiment of the method of this invention for fetching microprocessor's instructions. In Fig. 4, by taking advantage of an instruction reading-amount register **411**, which is set binary "1" for reading two instructions when a processing unit **410** has pre-read and pre-decoded a conditional branch instruction, namely, the method will read two instructions instead of one in the next instruction cycle. On the contrary, the instruction reading-amount register **411** is set binary "0" for reading one instruction when the processing unit **410** has pre-read the next instruction and found it in a form other than the conditional branch.

[0015] The program memory module of this invention is divided into an odd-page and an even-page program memory portion **407, 406**. As soon as an odd or an even address buffer register **405, 404** is enabled, the odd-page or the even-page program memory portion **407, 406** will be chosen and read by an instruction buffer register **409**. Regarding detailed operation, several examples are described below.

[0016] In the case the instruction reading-amount register is "0" while an address line set is "10".

As an incremental circuit **401** creates an address "11" and a selection switch "S" of a multiplexer **402, 403** is connected with the Least Significant Bit (LSB) of address lines or of the incremental circuit **401** respectively, hence the multiplexer **402** chooses the address "10" while the multiplexer **403** chooses the address "11". Then, the even-page address buffer register **404** is enabled via another multiplexer **414** to make the even-page program memory portion **406** readable for the instruction buffer register **409** to read the instruction address via a

multiplexer **408**.

In the case the instruction reading-amount register is "0" while the address line set is "11".

As the incremental circuit **401** creates an address "12", the multiplexer **402** chooses the address "12" while the multiplexer **403** chooses the address "11". Then, the odd address buffer register **405** is enabled via a multiplexer **413** to make the odd-page program memory portion **407** readable for the instruction buffer register **409** to read the instruction address via a multiplexer **408**.

In the case the instruction reading-amount register is "1" while the address line set is "11".

As the incremental circuit **401** creates an address "12", the multiplexer **402** chooses the address "12" while the multiplexer **403** chooses the address "11", and both the odd and the even address buffer registers **405**, **404** are enabled to make the odd-page and the even-page program memory readable. The address chosen by the instruction buffer register **409** is determined by the processing unit **410** (because the selection switch "S" of the multiplexer **412** is 1 depending on the processing unit **410**).

[0017] The operation manner of this invention is described below in connection with a program example shown in Fig. 5.

[0018] In processing a conditional branch instruction at PC address 10, the program pre-reads an unconditional branch instruction at PC address 11, and when processing the unconditional branch instruction, the program pre-reads an instruction at a next address. Referring to the timing chart of Fig. 5, when the instruction at address 9 is executed and when the pre-read next instruction at address 10 is decoded as a conditional branch instruction for example, the

instruction reading-amount register is set "1" such that the time the instruction at address 10 is executed, the instruction at address 11 and 12 are pre-read. If the instruction at address 11 is chosen and found by the processing unit as an unconditional branch instruction to be executed next, then the instruction reading-amount register is set "0". In the next instruction cycle, the instruction at address 11 is substituted by no operation (NOP). Then the instruction at a new address 100 is fetched and decoded as a general logic instruction and the instruction reading-amount register is set "0" for execution of that instruction corresponding to the address 100 in the next instruction cycle. Meanwhile, the next instruction at address 101 of the program memory is pre-read and decoded.

[0019] The program example in Fig. 6 is about the same with that in Fig. 5, except that the conditional branch instruction chosen this time is a "CALL" or a "RETURN" instruction at address 12. In the instruction cycle at the PC address 200, the instruction at address 12 is substituted by NOP, and the instruction at a new address 200 is fetched and decoded a general logic instruction. Thus, the instruction reading-amount register is set "0" and the instruction at the address 200 will be executed in the next instruction cycle. The instruction at address 201 is pre-read and interpreted a return instruction, and in the next instruction cycle, the instruction at address 201 is substituted by NOP, then the instruction at the return address 13 is read and decoded.

[0020] According to abovesaid, it is understood that reading two instructions is necessary only when a pre-read instruction is a conditional branch one, otherwise (about 80%) one instruction is needed to be read so as to save breath of reading unnecessary program memory for reducing power consumption.

[0021] In the above described, at least one preferred embodiment has been

described in detail with reference to the drawings annexed, and it is apparent that numerous variations or modifications may be made without departing from the true spirit and scope thereof, as set forth in the claims below.